APPEAL BRIEF UNDER 37 C.F.R. § 41.37

TABLE OF CONTENTS

PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of: Anil Seth et al.	Examiner: John J. Romano
Serial No.: 10/087,296	Group Art Unit: 2192
Filed: March 01, 2002	Docket: 1488.008US1

For: A TECHNIQUE FOR COMPILING COMPUTER CODE TO REDUCE ENERGY CONSUMPTION WHILE EXECUTING THE CODE

APPEAL BRIEF UNDER 37 CFR § 41.37

Mail Stop Appeal Brief- Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

The Appeal Brief is presented in support of the Notice of Appeal to the Board of Patent Appeals and Interferences, filed Herewith, from the Final Rejection of claims 1-44 of the above-identified application, as set forth in the Final Office Action mailed on January 3, 2007.

The Commissioner of Patents and Trademarks is hereby authorized to charge Deposit Account No. 19-0743 in the amount of $500.00 which represents the requisite fee set forth in 37 C.F.R. § 41.20(b)(2). The Appellants respectfully request reconsideration and reversal of the Examiner's rejections of pending claims.

1. REAL PARTY IN INTEREST

The real party in interest of the above-captioned patent application is the assignee, SASKEN COMMUNICATION TECHNOLOGIES LIMITED.

2. RELATED APPEALS AND INTERFERENCES

There are no other appeals or interferences known to Appellant that will have a bearing on the Board's decision in the present appeal.

3. STATUS OF THE CLAIMS

The present application was filed March 1, 2002 with claims 1-44. A non-final Office Action was mailed January 25, 2005. A Final Office Action was mailed October 20, 2005. Applicant filed an Request for Continued Examination on April 19, 2006. A non-final Office Action was mailed June 15, 2006. A Final Office Action was mailed January 3, 2007. Claims 1-44 stand twice rejected, remain pending, and are the subject of the present Appeal.

4. STATUS OF AMENDMENTS

No amendments have been made subsequent to the Final Office Action dated January 3, 2007.

5. SUMMARY OF CLAIMED SUBJECT MATTER

Independent Claim 1

Some aspects of the present inventive subject matter include, but are not limited to, as recited in independent claim 1, a method of compiling computer code including power-down instructions to reduce power consumption during execution of the code while satisfying user-specified real-time performance constraints on a microprocessor. (p. 5, lines 5-8; FIG. 23, No. 2302). The method identifies one or more potential locations in the computer code where the power-down instructions can be inserted (p. 6, lines 6-8; FIG. 1, No. 110), selects locations to insert the power-down instructions from the identified potential locations in the code based on reducing power consumption and satisfying user-specified real-time performance constraints (p. 12, lines 8-13, FIG. 1, No. 150), and inserts the power-down instructions in the selected locations to reduce the power consumption during the execution of the code while satisfying user-specified real-time performance constraints (p. 2, lines 21-26; FIG. 1, No. 160).

Independent Claim 14

In another embodiment, as recited in independent claim 14, a computer-readable medium (FIG. 23, No. 2304) having computer-executable instructions for reducing power consumption while running a computer program (p. 2, lines 27-29) identifies one or more potential locations in the computer program where power-down instructions can be inserted (p. 6, lines 6-8; FIG. 1, No. 110), selects locations to insert the power-down instructions from the identified potential locations in the program based on satisfying user-specified real-time performance constraints (p. 12, lines 8-13; FIG. 1, No. 150), and inserts the power-down instructions in the selected locations to reduce power consumption while running the computer program while satisfying the user-specified real-time performance constraints (p. 2, lines 21-26; FIG. 1, No. 160).

Independent Claim 24

In another embodiment, as recited in independent claim 24, a computer system (p. 3, line 7; FIG. 23, No 2300) for reducing power consumption during execution of computer code includes a storage device (FIG. 23, No. 2304), an output device (FIG. 23, No. 2318), and a processor (FIG. 23, No. 2302). The processor is programmed to repeatedly perform the steps of identifying one or more potential locations in the computer code where power-down instructions can be inserted (p. 6, lines 6-8; FIG. 1, No. 110), selecting locations to insert the power-down instructions from the identified potential locations in the code based on satisfying user-specified real-time performance constraints (p. 12, lines 8-13; FIG. 1, No. 150), and inserting the power-down instructions in the selected locations to reduce power consumption during the execution of the code while satisfying the user-specified real-time performance constraints (p 2, lines 21-26; FIG. 1, No. 160).

Independent Claim 34

In another embodiment, as recited in independent claim 34, a computer-readable medium (FIG. 23, No. 2304) includes a computer program having instructions for causing a computer to perform a method of selectively controlling power to different functional units of the computer (p. 12, lines 8-13; FIG. 1, No. 150). The instructions include power-down instructions inserted in the computer-program in selected locations based on reducing power consumption and satisfying user-specified real-time performance constraints (p. 2, lines 21-26; FIG. 1, No. 160). The power-down instructions in the selected locations reduce the power consumption during the execution of the code while satisfying the user-specified real-time performance constraints (p. 2, lines 21-26; FIG. 1, No. 160).

This summary does not provide an exhaustive or exclusive view of the present subject matter, and Appellant refers to the appended claims and its legal equivalents for a complete statement of the invention.

6. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

Claims 1, 2, 11-15, 22-25, 32-36, 43 and 44 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Bartley (U.S. Patent No. 6,219,796) in view of Y. Li et al. ("A framework for estimating and minimizing energy dissipation of embedded hw/sw systems").

Claims 3-10, 16-21, 26-31 and 37-42 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Bartley (U.S. Patent No. 6,219,796) in view of Y. Li et al. ("A framework for estimating and minimizing energy dissipation of embedded hw/sw systems") and further in view of G. Ramalingam ("Data Flow Frequency Analysis," SIGPLAN Conference on Programming Language Design and Implementation, 1996).

7. ARGUMENT

A) The Applicable Law under 35 U.S.C. §103(a)

The Patent Office bears the initial burden of factually supporting a *prima facie* case of obviousness. (MPEP 2142). In order for the Office Action to establish a *prima facie* case of obviousness, three criteria must be met. First, there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or to combine reference teachings. Second, there must be a reasonable expectation of success. Finally, the prior art reference (or references when combined) must teach or suggest all the claim limitations. The teaching or suggestion to make the claimed combination and the reasonable expectation of success must both be found in the prior art, and not based on applicant's disclosure. (MPEP § 2142 (citing *In re Vaeck,* 947 F.2d 488, 20 USPQ2d 1438 (Fed. Cir. 1991))).

B) Discussion of the rejection of claims 1, 2, 11-15, 22-25, 32-36, 43 and 44 under 35 U.S.C. § 103(a) as being unpatentable over Bartley (U.S. Patent No. 6,219,796) in view of Y. Li et al. ("A framework for estimating and minimizing energy dissipation of embedded hw/sw systems").

The Final Office Action rejected claims 1, 2, 11-15, 22-25, 32-36, 43 and 44 under 35 USC § 103(a) as being unpatentable over Bartley (U.S. Patent No. 6,219,796) in view of Y. Li et al. (A framework for estimating and minimizing energy dissipation of embedded hw/sw systems). The Applicant respectfully appeals and seeks reversal of this rejection.

The Final Office Action errs in at least two areas. First, the Final Office Action misinterprets the content and import of Bartley and Li. Specifically, the Applicant respectfully submits that neither reference discloses the use of power down instructions based on user-specified real-time performance constraints. The Applicant further respectfully submits that Li does not even relate to the use of power down instructions, but rather relates only to a very specific instance of hardware modifications, and as such, Li is not properly combinable with Bartley. Second, the Final Office Action fails to see the contribution to the art of the claimed invention. The presently claimed invention contributes to the art in that there may be many places in code where a power down instruction could be added, and the claimed invention allows one to determine where to put those power down instructions to optimize power consumption within user specified constraints to make sure that the overall program performance is as desired. Neither reference, alone or combined, provides this capability.

Claim 1 recites a method of compiling computer code including power-down instructions to reduce power consumption during execution of the code while satisfying user-specified real-time performance constraints on a microprocessor. One or more potential locations are identified in the computer code where the power-down instructions can be inserted. Locations to insert the power-down instructions are selected from the identified potential locations in the code based on reducing power consumption and satisfying user-specified real-time performance constraints. The power-down instructions are inserted in the selected locations to reduce the power consumption during the execution of the code while satisfying user-specified real-time performance constraints.

As indicated above in claim 1, the present claims allow a tradeoff between performance and power conservation based on user specified constraints for execution of program code. The claims clearly point out reducing power consumption during the execution of the code while satisfying user-specified performance constraints and clearly point out the difference between the claimed invention and the cited references. The references used to reject the claims either focus entirely on efficient use of power or selection of hardware sizes to accomplish performance and energy usage goals. Neither reference, alone or combined, teach or suggest the use of power down instructions in code to reduce power consumption while satisfying user specified performance constraints.

The references will now be discussed in further detail to more clearly point out the differences between the references, alone or combined, and the claims.

Bartley describes scanning of code to determine that a functional unit of a processor will not be used during execution of a program. That functional unit is then shut down to conserve power. Instructions may be inserted to turn such functional units off and on. Thresholds may be used to make sure that at least some benefit is provided by doing so. As noted several times in Col. 7, Bartley focuses on ensuring efficient use of power. Lines 17-18, and 38. The entire context of Col. 7 is to efficiently use power. Despite references to static and dynamic program analysis, and thresholds, they all relate to identifying sections of code where a shutdown may occur to efficiently use power, and do not relate to code performance. No mention is made of code performance in Bartley, only efficient use of power.

Y. Li et al. is used to modify hardware, such as cache and main memory size, to trade off performance and energy use goals. It has nothing to do with inserting power down instructions in code. It is a very inflexible application of power conservation techniques to an embedded and dedicated hardware platform. Once done, it is optimized to an individual application. Thus, it teaches away from the ability, or even any desire to use power control instructions in programs. Neither reference, alone or combined teaches or suggests inserting power down instructions to reduce power consumption while satisfying user-specified real-time constraints.

The Final Office Action states on page 10 that Bartley does not disclose "…satisfying user-specified real-time constraints..", but then contends on page 11 that Li et al., discloses "…satisfying user-specified real-time constraints…". The Office Action then argues that it would be obvious to combine Bartley and Li et al. Applicant does not believe that the references are properly combinable, as each is directed to very different aspects of power reduction.

Bartley inserts power-down instructions into programming with the goal of reducing power consumption. Li et al., describes a very different type of system. In Li et al., an embedded system is described, where the software and hardware components are designed and modified with power conservation in mind. Software may be transformed, and different sizes of cache and main memory are considered to optimize power conservation. In that process, which is very different from the power down aspects of the present application and Bartley, there is no consideration of powering down different components. Rather, the components themselves may be modified in size to conserve power.

As can be seen, the approaches of Li et al. versus those of the present application and Bartley are very different. While both may be directed to improving energy consumption, there is nothing in Bartley or Li et al. that indicates different aspects of them may be combined. The Final Office Action first contends that the motivation to combine them comes from Bartley, "as he refers to program segments having a duration longer than a "predetermined threshold" (Column 7, lines 42-43) , wherein it is obvious the threshold may be determined by a user either via a user selected algorithm or other user input." The purported motivation is in the context of finding code segments of long enough duration to make it worth shutting down a functional unit. If it would take longer than the amount of time required for execution of the segment to turn it off and then turn it back on, it would not make sense to turn it off in the first place. "Various power modeling techniques can be used to determine the length of time during which it is more efficient to turn a component off (or partially off) then on again versus leaving it on." Col. 7, lines 16-19. It does not relate directly to satisfying user-specified real time constraints or program performance as currently claimed. As such, it would not suggest to one of skill in the art that performance optimization goals should be considered. In practice, with the presently claimed invention, there may be many places in code where a power down instruction could be added. The claimed invention allows one to determine where to put them to optimize power consumption within user specified constraints.

Furthermore, in response to the contention on page 11 of the Final Office Action that Li et al. discloses "...satisfying user-specified real-time constraints...", it should be noted that Li et al. describes different optimization goals in the context of changing sizes of cache and main memory, not in the context of powering down different functional units. Thus, it is not proper to ascribe the performance constraints in this context with the insertion of power conservation instructions. One relates to hardware design, and the other relates to programming existing hardware. This great difference in architecture and methodology of conserving power makes it highly unlikely that Li et al. would be considered by one of skill in the art when focusing on powering down different functional units. It also places the likelihood of success of such a combination in great jeopardy.

Finally, the motivation to combine Bartley and Li et al. cited on page 11 of the Final Office Action is not in accordance with the law. The Final Office Action states that "the motivation is disclosed by Bartley, as he refers to program segments having a duration longer than a predetermined threshold." However, the Final Office Action fails to explain the connection between identifying code segments relating to a functional unit that have a duration longer than a predetermined threshold (which is not user-specified, but rather an inherent aspect of the program code and the associated functional unit), and selecting locations to insert the power-down instructions based on reducing power consumption and satisfying user-specified real-time performance constraints. Moreover, the Final Office Action states on page 11 that it is "obvious the threshold may be determined by a user either via a user selected algorithm or other user input." Once again, there is no relation between the determination of a threshold relating to the duration of a program segment for a related functional unit, and the insertion of power-down instructions that satisfy user-specified real-time performance constraints.

Further, as the time threshold is related to determining whether there would be any benefit obtained by powering down, it is clearly not a user specified real-time constraint. Specifically, the Applicant respectfully submits that the threshold of Bartley is determined in the following manner, which is clearly not user specified: "Various power modeling techniques can be used to determine the length of time during which it is more efficient to turn a component off (or partially off) then on again versus leaving it on." Col. 7, lines 16-19. Each threshold appears to be fixed and based on efficiency, not user specified time constraints. Thus, the claim language of inserting power-down instructions while satisfying user-specified real-time constraints does not necessarily flow from the cited language of Bartley, and the rejection should be withdrawn, as at least one element of the claims is lacking from the combination even if proper.

Independent claims 14, 24, and 34 all contain similar recitations and distinguish the references for at least the same reasons.

Claims 3-10, 16-21, 26-31 and 37-42 were rejected under 35 USC § 103(a) as being unpatentable over Bartley (U.S. Patent No. 6,219,796) in view of Y. Li et al. and further in view of G. Ramalingam (Data Flow Frequency Analysis, SIGPLAN Conference on Programming Language Design and Implementation, 1996). This rejection is respectfully traversed, as all depend from independent claims that are believed allowable.

Thus, because neither Bartley nor Li disclose the use of power down instructions based on user-specified real-time performance constraints, the cited references do not disclose each and every element of the claims. Further, because Li does not relate to the use of power down instructions (but rather to hardware modifications), Li is not properly combinable with Bartley.

Further Arguments Regarding Claims 11, 12, 22, 32, and 43

In addition to the arguments put forth above for claims 11, 12, 22, 32, and 43, the Applicant further puts forth the following arguments separately for these claims.

The Final Office Action contends on pages 11-13 that Li et al. discloses in its section 5.2 and its Table 2 a "number of power-down instructions that can be inserted in an execution path, including one or more identified potential locations" and a "number of additional cycles of execution time the user is willing to incur due to an insertion of the power-down instruction at each of the identified potential locations." The Applicant respectfully disagrees. As pointed out above, Li et al. simply does not deal with power-down instructions at all. Section 5.2 and Table 2 of Li et al. similarly do not mention the use of power down instructions. Rather, Table 2 appears to relate to different software applications (*i.e.,* bsort, eg2, ismooth, and itimp), and the optimization of architecture with those applications. So, for these additional reasons, the rejection of claims 11, 12, 22, 32, and 43 under 103(a) is improper, and the rejection should be reversed.

Moreover, there is a distinct difference between Bartley and claims 11, 12, 22, 32, and 43. Bartley relates to identifying program segments of a processor during which a functional unit of a processor is not used. Bartley then determines whether it is worthwhile to place a power down instruction into the code based on the duration of time that the functional unit is not used. The user-specified number of power-down instructions and additional cycles of execution time recited in these claims relate to the extra execution time of the processor caused by the addition of the power down instructions. There is very little relation between Bartley's time period that a functional unit is ***not*** used, and the number of processor cycles ***added*** by power down instructions. Therefore, the Final Office Action's statement on page 12 in its rejection of claim 11 that Bartley's teaching of a time threshold (the duration that a functional unit is not being used) serves as a motivation for someone adding power down instructions to code to consider the additional execution cycles of the power down code is a *non-sequitur,* and fails to establish a *prima facie* case of obviousness.

A further distinction includes the lack of concern in Bartley for real time performance of the executing code. The claims clearly indicate reduction of power consumption while satisfying real time performance constraints related to execution of the code. Bartley uses a threshold to determine whether sufficient power would be saved, not whether performance of code constraints are met.

8. SUMMARY

For the reasons argued above, the pending claims were not properly rejected under § 103(a). It is respectfully submitted that the references of record do not render the claims obvious and that the claims are patentable over the cited references. Reversal of the rejection and allowance of the pending claims are respectfully requested.

Respectfully submitted,

ANIL SETH et al.

By their Representatives,

SCHWEGMAN, LUNDBERG, WOESSNER & KLUTH, P.A.
P.O. Box 2938
Minneapolis, MN 55402

Date March 9 2007 By [illegible]
David D'Zurilla
Reg. No. 36,776

CERTIFICATE UNDER 37 CFR 1.8: The undersigned hereby certifies that this correspondence is being filed using the USPTO's electronic filing system EFS-Web, and is addressed to: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450 on this 7 day of March 2007.

John D. Gustav-Wrathall
Name

[illegible]
Signature

CLAIMS APPENDIX

1. A method of compiling computer code including power-down instructions to reduce power consumption during execution of the code while satisfying user-specified real-time performance constraints on a microprocessor, comprising:

identifying one or more potential locations in the computer code where the power-down instructions can be inserted;

selecting locations to insert the power-down instructions from the identified potential locations in the code based on reducing power consumption and satisfying user-specified real-time performance constraints; and

inserting the power-down instructions in the selected locations to reduce the power consumption during the execution of the code while satisfying user-specified real-time performance constraints.

2. The method of claim 1, wherein the code is written for a microprocessor having distinct functional units.

3. The method of claim 2, wherein identifying potential locations comprises:

identifying potential locations based on the functional units not being used in the potential locations, wherein the functional units not being used are determined based on functional unit usage transfer functions at each of the potential locations as specified in standard monotone data-flow frameworks.

4. The method of claim 3, wherein identifying potential locations is accomplished by statically analyzing processor cycles prior to executing the code.

5. The method of claim 4, wherein statically analyzing processor cycles is accomplished by statically analyzing the text in the code for the functional units not being used prior to executing the code.

6. The method of claim 3, wherein each of the power-down instructions comprise:

a first power-down instruction operable to reduce power to all of the at least one functional unit, such that the functional unit is placed in a low state of readiness and a second power-down instruction operable to reduce power to only a part of the at least one functional unit, such that the functional unit is placed in an intermediate state of readiness.

7. The method of claim 1, wherein selecting identified potential locations on the computer code based on satisfying the user-specified real-time performance constraints, comprise:

executing the code to generate power-profiling information associated with each of the identified potential locations;

executing the code to generate execution path-profiling information associated with each of the identified potential locations;

assigning a weight factor to each of the identified potential locations based on the generated power-profiling and path-profiling information; and

selecting the locations to insert the power-down instruction from the identified locations based on the assigned weight factors and the user-specified real-time performance constraints.

8. The method of claim 7, wherein executing the code to generate path-profiling information to each of the identified potential locations further comprises:

generating execution probability of each of the identified potential locations based on the generated path-profiling information.

9. The method of claim 8, wherein assigning the weight factor comprises:

extracting potential energy savings for each of the identified potential locations using the generated power profile analysis information; and

assigning the weight factor to each of the identified potential locations based on the extracted potential energy savings and the generated execution probability.

10. The method of claim 9, wherein assigning the weight factor further comprises:

executing the code to assign a first weight factor based on the extracted potential energy savings to each of the identified potential locations;

executing the code to assign a second weight factor based on execution probability at each of the identified potential locations;

computing a product of the first and second weight factors for each of the identified potential locations;

calculating the weight factor for each of the identified potential locations based on the computed product of the first and second weight factors; and

assigning the calculated weight factor to each of the identified potential locations.

11. The method of claim 1, wherein user-specified real-time constraints comprise:

the number of power-down instructions that can be inserted in an execution path, including one or more identified potential locations.

12. The method of claim 11, wherein user-specified real-time performance constraints comprise:

the number of additional cycles of execution time the user is willing to incur due to an insertion of the power-down instruction at each of the identified potential locations.

13. The method of claim 11, further comprising:

inserting power-up instruction in the code to restore at least one functional unit to a ready state powered-down by the inserted power-down instructions.

14. A computer-readable medium having computer-executable instructions for reducing power consumption while running a computer program, comprising:

identifying one or more potential locations in the computer program where power-down instructions can be inserted;

selecting locations to insert the power-down instructions from the identified potential locations in the program based on satisfying user-specified real-time performance constraints; and inserting the power-down instructions in the selected locations to reduce power consumption while running the computer program while satisfying the user-specified real-time performance constraints.

15. The medium of claim 14, wherein the code is written for a microprocessor including distinct functional units.

16. The medium of claim 14, wherein identifying potential locations comprises:

identifying the potential locations based on the functional units not being used in the potential locations, wherein the functional units not being used are determined based on functional unit usage transfer functions at each of the potential locations as specified in standard monotone data-flow frameworks.

17. The medium of claim 16, wherein identifying potential locations is accomplished by statically analyzing processor cycles prior to running the program.

18. The medium of claim 14, wherein selecting the identified potential locations on the computer program based on satisfying the user-specified real-time constraints, comprise:

running the computer program to generate power-profiling information associated with each of the identified potential locations;

running the computer program to generate execution path-profiling information associated with each of the identified potential locations;

assigning a weight factor to each of the identified potential locations based on the generated power-profiling and path-profiling information; and

selecting the locations to insert the power-down instructions from the identified locations based on the assigned weight factors and the user-specified real-time performance constraints.

19. The medium of claim 18, wherein running the program to generate path- profiling information to each of the identified potential locations further comprises:

generating running probability of each of the identified potential locations based on the generated path-profiling information.

20. The medium of claim 19, wherein assigning the weight factor comprises:

extracting potential energy savings for each of the identified potential locations using the generated power profile analysis information; and

assigning the weight factor to each of the identified potential locations based on the extracted potential energy savings and the generated running probability.

21. The medium of claim 20, wherein assigning the weight factor further comprises:

running the program to assign a first weight factor based on the extracted potential energy savings to each of the identified potential locations;

running the program to assign a second weight factor based on execution probability at each of the identified potential locations;

computing a product of the first and second weight factors for each of the identified potential locations;

calculating the weight factor for each of the identified potential locations based on the computed product of the first and second weight factors; and

assigning the calculated weight factor to each of the identified potential locations.

22. The medium of claim 14, wherein user-specified real-time performance constraints comprise:

the number of power-down instructions that can be inserted in a running path including one or more identified potential locations.

23. The medium of claim 22, further comprising:

inserting power-up instructions in the program to restore at least one functional unit to a ready state powered-down by the inserted power-down instructions.

24. A computer system for reducing power consumption during execution of computer code, comprising:

a storage device;

an output device; and

a processor programmed to repeatedly perform a method, comprising:

identifying one or more potential locations in the computer code where power-down instructions can be inserted;

selecting locations to insert the power-down instructions from the identified potential locations in the code based on satisfying user-specified real-time performance constraints; and

inserting the power-down instructions in the selected locations to reduce power consumption during the execution of the code while satisfying the user-specified real-time performance constraints.

25. The system of claim 24, wherein the code is written for a microprocessor including distinct functional units.

26. The system of claim 24, wherein identifying the potential locations comprises:

identifying the potential locations based on the functional units not being used in the potential locations, wherein the functional units not being used are determined based on functional unit usage transfer functions at each of the potential locations as specified in standard monotone data-flow frameworks.

27. The system of claim 26, wherein identifying the potential locations is accomplished by statically analyzing processor cycles prior to executing the code.

28. The system of claim 24, wherein selecting the identified potential locations on the computer code based on satisfying the user-specified real-time performance constraints, comprises:

executing the code to generate power-profiling information associated with each of the identified potential locations;

executing the code to generate execution path-profiling information associated with each of the identified potential locations;

assigning a weight factor to each of the identified potential locations based on the generated power-profiling and path-profiling information; and

selecting the locations to insert the power-down instruction from the identified locations based on the assigned weight factors and the user-specified real-time performance constraints.

29. The system of claim 28, wherein executing the code to generate path-profiling information to each of the identified potential locations further comprises:

generating execution probability of each of the identified potential locations based on the generated path-profiling information.

30. The system of claim 29, wherein assigning the weight factor comprises:

extracting potential energy savings for each of the identified potential locations using the generated power profile analysis information; and

assigning the weight factor to each of the identified potential locations based on the extracted potential energy savings and the generated execution probability.

31. The system of claim 30, wherein assigning the weight factor further comprises:

executing the code to assign a first weight factor based on the extracted potential energy savings to each of the identified potential locations;

executing the code to assign a second weight factor based on execution probability to each of the identified potential locations;

computing a product of the first and second weight factors for each of the identified potential locations;

calculating the weight factor for each of the identified potential locations based on the computed product of the first and second weight factors; and

assigning the calculated weight factor to each of the identified potential locations.

32. The system of claim 24, wherein user-specified real-time performance constraints comprise:

the number of power-down instructions that can be inserted in an execution path including one or more identified potential locations.

33. The system of claim 32, further comprising:

inserting power-up instructions in the code to restore at least one functional unit to a ready state powered-down by the inserted power-down instructions.

34. A computer-readable medium having a computer program including instructions for causing a computer to perform a method of selectively controlling power to different functional units of the computer, the instructions comprising:

power-down instructions inserted in the computer-program in selected locations based on reducing power consumption and satisfying user-specified real-time performance constraints; and

wherein the power-down instruction in the selected locations reduce the power consumption during the execution of the code while satisfying the user-specified real-time performance constraints.

35. The medium of claim 34, wherein inserting power-down instructions in the computer-program in selected locations further comprises:

identifying one or more potential locations in the computer program where power-down instructions can be inserted;

selecting locations to insert the power-down instructions from the identified potential locations in the program based on satisfying user-specified real-time performance constraints; and

inserting the power-down instructions in the selected locations to reduce power consumption while running the computer program while satisfying the user-specified real-time performance constraints.

36. The medium of claim 35, wherein the code is written for a microprocessor including distinct functional units.

37. The medium of claim 35, wherein identifying potential locations comprises:

identifying the potential locations based on the functional units not being used in the potential locations, wherein the functional units not being used are determined based on functional unit usage transfer functions at each of the potential locations as specified in standard monotone data-flow frameworks.

38. The medium of claim 37, wherein identifying potential locations is accomplished by statically analyzing processor cycles prior to running the program.

39. The medium of claim 35, wherein selecting the identified potential locations on the computer program based on satisfying the user-specified real-time constraints, comprise:

running the computer program to generate power-profiling information associated with each of the identified potential locations;

running the computer program to generate execution path-profiling information associated with each of the identified potential locations;

assigning a weight factor to each of the identified potential locations based on the generated power-profiling and path-profiling information; and

selecting the locations to insert the power-down instructions from the identified locations based on the assigned weight factors and the user-specified real-time performance constraints.

40. The medium of claim 39, wherein running the program to generate path- profiling information to each of the identified potential locations further comprises:

generating running probability of each of the identified potential locations based on the generated path-profiling information.

41. The medium of claim 40, wherein assigning the weight factor comprises:

extracting potential energy savings for each of the identified potential locations using the generated power profile analysis information; and assigning the weight factor to each of the identified potential locations based on the extracted potential energy savings and the generated running probability.

42. The medium of claim 41, wherein assigning the weight factor further comprises:

running the program to assign a first weight factor based on the extracted potential energy savings to each of the identified potential locations;

running the program to assign a second weight factor based on execution probability at each of the identified potential locations;

computing a product of the first and second weight factors for each of the identified potential locations;

calculating the weight factor for each of the identified potential locations based on the computed product of the first and second weight factors; and

assigning the calculated weight factor to each of the identified potential locations.

43. The medium of claim 35, wherein user-specified real-time performance constraints comprise:

the number of power-down instructions that can be inserted in a running path including one or more identified potential locations.

44. The medium of claim 43, further comprising:

inserting power-up instructions in the program to restore at least one functional unit to a ready state powered-down by the inserted power-down instructions.

EVIDENCE APPENDIX

None.

RELATED PROCEEDINGS APPENDIX

None.